Chapter 6
Ripple Effect in Web Applications

Nashat Mansour
American University of Beirut, Lebanon

Nabil Baba
Lebanese American University, Lebanon

ABSTRACT

The number of internet web applications is rapidly increasing in a variety of fields and not much work has been done for ensuring their quality, especially after modification. Modifying any part of a web application may affect other parts. If the stability of a web application is poor, then the impact of modification will be costly in terms of maintenance and testing. Ripple effect is a measure of the structural stability of source code upon changing a part of the code, which provides an assessment of how much a local modification in the web application may affect other parts. Limited work has been published on computing the ripple effect for web application. In this paper, the authors propose, a technique for computing ripple effect in web applications. This technique is based on direct-change impact analysis and dependence analysis for web applications developed in the.Net environment. Also, a complexity metric is proposed to be included in computing the ripple effect in web applications.

1. INTRODUCTION

Web applications and regular software differ from each other in several ways. They differ in the inclusion of complex multi-tiered, heterogeneous architecture including web applications and database servers. The boundaries of a web application in web platforms are the boundaries of the World Wide Web itself. Unlike normal software environment in regular applications, web applications are subject to more challenges. One challenge is that of determining and controlling the propagation of change in the web application that arises from a local modification.

DOI: 10.4018/978-1-4666-0023-2.ch006

Ripple effect measure has been identified as an important measure of change propagation. Imagine a stone being thrown into a pond: it makes a sound as it enters the water and causes ripples to move outwards. Transferring this image into a source code is easy. The stone entering the water is now the change to the source code of a program, the effect of the change ripples across the source code via data and control flow. The ripple effect reflects how possible it is that a change to a particular form, method or class is going to adversely affect the rest of a program. It helps in determining the scope of the change and presents a measure of the program complexity. It can also be considered an indicator of the stability of a program. Ripple effect was one of the earliest metrics concerned with the structure of a system and how its modules interact (Shepperd, 1993). Ripple effect can show the maintainer what the effect of a local change will be on the rest of the web application. It can highlight the fragile parts of the web application to be restructured for better stable performance.

A method for computing ripple effect was developed early by Yau et al. (1978). Tools have been developed to produce ripple effect measures for procedural software using Yau et al.'s algorithm which is based on set theory. Black (2001) reformulated the ripple effect computation using matrix arithmetic and used this formulation to produce a software tool, REST. However, Black's algorithm computes the ripple effect only for procedural programs. Mansour and Salem (2006) proposed ripple effect techniques for object oriented (OO) program based on an analysis of object-oriented dependencies, relations, and propagations inside and outside classes. Elish and Rine (2003, 2005) studied structural and logical stability of object oriented design. They assessed the capability of OO design stability indicators to be used as predictors of measures of the metrics of a stability metrics suite. Mattsson and Bosch (2000) assessed stability of evolving OO frameworks. Grosser et al. (2002) used case-based reasoning to predict software stability. Jazayeri (2002) applied retrospective analysis to successive software releases to evaluate its architectural stability with size and coupling measures. Further work on change impact analysis for object oriented programs can be found in: Rajlich (2001), Ryder and Tip (2001), Briand, Labiche, and Soccar (2002), Kung et al. (1994), and Lee, Offutt, and Alexander (2000).

In this paper we propose a technique for computing the ripple effect and logical stability for web applications focusing on.Net environment (ASP. Net and VB.Net) and using matrix arithmetic. Thus, the main contribution is in addressing the distinctive features of web application software. The computation of the ripple effect of a.Net web application is based on an analysis of ASP. Net and VB.Net (Bell, 2002) object-oriented dependencies, relations, and propagations locally and globally for ASP.Net (.aspx pages) and inside and outside classes for VB.Net and VBScripts. For brevity, this analysis is omitted herein and can be found in (Baba, 2007). Also, we propose a complexity metric for ASP.Net and VB.Net code to be included in computing the ripple effect. We compute the ripple effect for ASP.Net and VB.Net object-oriented at the code level. Both global and Form-Scope Propagation for.aspx pages (ASP. Net) and inter-class propagation and intra-class propagation for each class are considered. We also compute the architectural ripple effect at the system level. Each matrix used within the algorithm holds a particular type of information about the software under study.

This paper is organized as follows. Sections 2-7 describe our technique for computing the ripple effect and illustrates it with a running example. Section 8 concludes the paper.

2. WEB CODE COMPLEXITY METRIC

In this section, we propose a complexity metric for a.aspx page that contains no scripts, which is also applicable for.aspx.vb files. This metric is